

**AMENDMENTS TO THE CLAIMS**

Please amend the claims as follows.

1. – 11. (Cancelled)

12. (New) A method for verifying execution of a program, wherein the program comprises a first code portion and a second code portion, the method comprising:

- entering the first code portion, wherein the first code portion comprises a first plurality of instructions;
- executing the first code portion;
- calculating a first checksum during the execution of the first code portion, wherein the first checksum is calculated using information associated with at least one of the first plurality of instructions;
- comparing the first checksum to a first pre-calculated checksum prior to exiting the first code portion; and
- exiting the first code portion and entering the second code portion if the first checksum equals the first pre-calculated checksum.

13. (New) The method of claim 12, further comprising:

- entering the second code portion, wherein the second code portion comprises a second plurality of instructions;
- executing the second code portion;
- calculating a second checksum during the execution of the second code portion, wherein the second checksum is calculated using information associated with at least one of the second plurality of instructions;
- comparing the second checksum to a second pre-calculated checksum prior to exiting the second code portion; and
- exiting the second code portion if the first checksum equals the first pre-calculated checksum.

14. (New) The method of claim 12, further comprising:

detecting an anomaly if the first checksum equals the first pre-calculated checksum.

15. (New) The method of claim 12, wherein a last instruction in the first plurality of instructions to be executed prior to exiting the first code portion is modified to performing comparing the first checksum to a first pre-calculated checksum.
16. (New) The method of claim 12, wherein comparing the first checksum to a first pre-calculated checksum is performed after all of the first plurality of instructions have been executed.
17. (New) The method of claim 12, wherein the information associated with the at least one of the first plurality of instructions comprises at least one selected from the group consisting of content of the at least one of the first plurality of instructions, a type of the at least one of the first plurality of instructions, a function performed by the at least one of the first plurality of instructions, and a result generated by executing of the at least one of the first plurality of instructions.
18. (New) The method of claim 12, wherein the first code portion is bounded by at least one pair selected from the group consisting of an entry point and an exit point, a first jump address and a second jump address, a first branch jump and a second branch jump, a routine call and a corresponding return instruction, entry to interruption handling and exit from interruption handling.
19. (New) The method of claim 12, wherein the first pre-calculated checksum is calculated during compilation of the program.
20. (New) An electronic module, comprising:
  - a program, comprising a first code portion and a second code portion;
  - a first pre-calculated checksum;
  - a processor configured to execute the program, wherein executing the program comprises:
    - entering the first code portion, wherein the first code portion comprises a first plurality of instructions;
    - executing the first code portion;

calculating a first checksum during the execution of first code portion, wherein the first checksum is calculated using information associated with at least one of the first plurality of instructions;  
comparing the first checksum to the first pre-calculated checksum prior to exiting the first code portion; and  
exiting the first code portion and entering the second code portion if the first checksum equals the first pre-calculated checksum.

21. (New) A card comprising the electronic module of claim 20.
22. (New) A method for verifying execution of a program, wherein the program comprises a first routine and a second routine, the method comprising:  
entering the first routine, wherein the first routine comprises a plurality of instructions and each of the plurality of instructions is associated with a value;  
initializing a counter associated with the first routine, prior to executing the first routine;  
executing the first routine, wherein the counter is incremented by the value associated with each of the plurality of instructions executed during the execution of the first routine;  
comparing a value of the counter to a pre-calculated value prior to exiting the first routine;  
exiting the first routine and entering the second routine if the value of the counter equals the pre-calculated value.
23. (New) The method of claim 22, further comprising:  
detecting an anomaly if the value of the counter is not equal to the pre-calculated value.
24. (New) The method of claim 22, wherein the first routine comprises a first branch and a second branch and wherein the value of the counter resulting from executing instructions in the first branch is equal to the value of the counter resulting from executing instructions in the second branch.
25. (New) The method of claim 22, wherein the value associated with each of the plurality of instructions is unique.

26. (New) The method of claim 22, wherein the value associated with a first one of the plurality of instructions is the same as the value associated with a second one of the plurality of instructions, if a type of the first one of the plurality of instructions is the same as a type of the second one of the plurality of instructions.
27. (New) An electronic module, comprising:
- a program, comprising a first routine and a second routine;
  - a value of a first pre-calculated counter;
  - a processor configured to execute the program, wherein executing the program comprises:
    - entering the first routine, wherein the first routine comprises a plurality of instructions and each of the plurality of instructions is associated with a value;
    - initializing a counter associated with the first routine, prior to executing the first routine;
    - executing the first routine, wherein the counter is incremented by the value associated with each of the plurality of instructions executed during the execution of the first routine;
    - comparing a value of the counter to a pre-calculated value prior to exiting the first routine;
    - exiting the first routine and entering the second routine if the value of the counter equals the pre-calculated value.
28. (New) A card comprising the electronic module of claim 27.